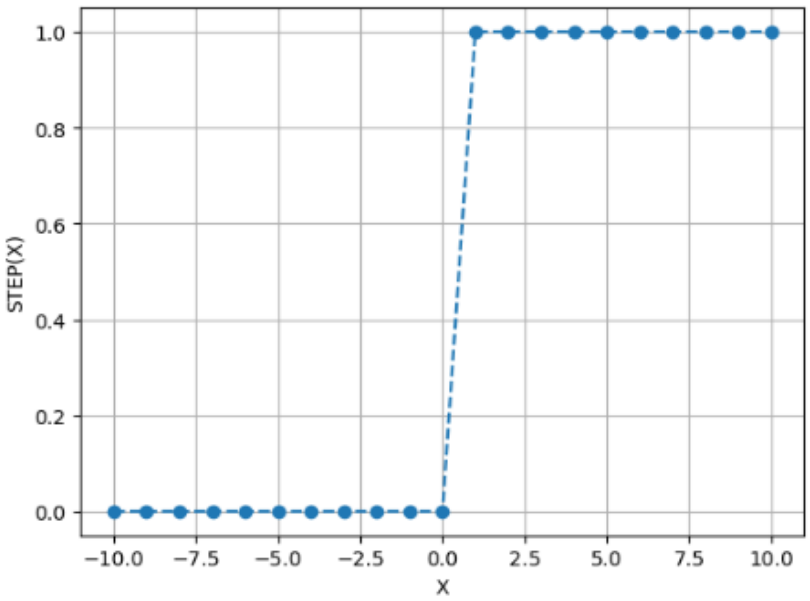
Activation Functions

1. **Step Function:**

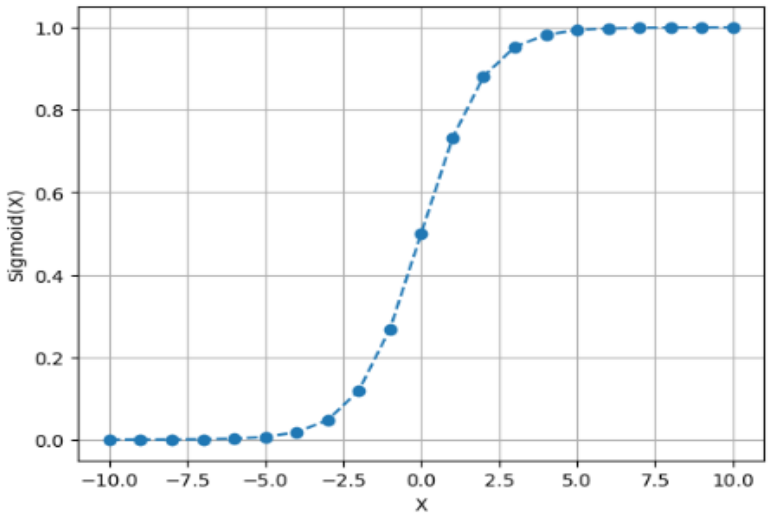
```
y = list(map(lambda n: 1 if n>0.5 else 0, x))
plot_graph(y,"STEP(X)")
```



- Advantages: Simple, easy to implement and interpret.
- Disadvantages: Output is not continuous and cannot be differentiated. Limited to binary classification problems.

2. **Sigmoid Function:**

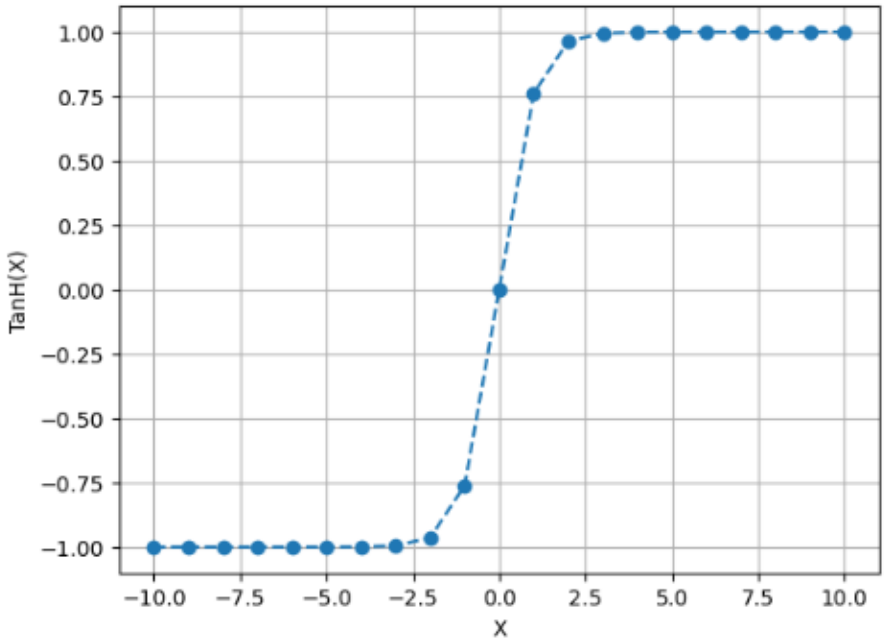
[5]:
```
y = 1 / (1 + np.exp(-x))
plot_graph(y, "Sigmoid(X)")
```



- Advantages: Output is bounded between 0 and 1 which makes it suitable for binary classification problems. It is differentiable and used in backpropagation.
- Disadvantages: Can suffer from vanishing gradients and prone to saturation for extreme values. Not zero-centered.

3. Tanh Function:

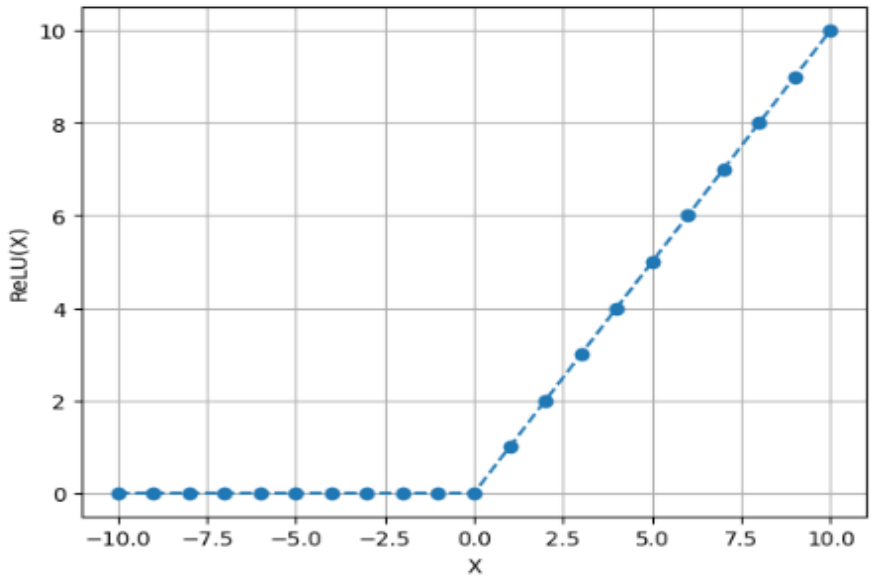
```
6]:
y = (np.exp(2*x) - 1) / (np.exp(2*x) + 1)
plot_graph(y, "TanH(X)")
```



- Advantages: Similar to sigmoid but zero-centered. Outputs are bound between -1 and 1.
- Disadvantages: Like sigmoid, it can suffer from vanishing gradients.

4. ReLU Function:

```
y = list(map(lambda a: a if a>=0 else 0, x))
plot_graph(y,"ReLU(X)")
```



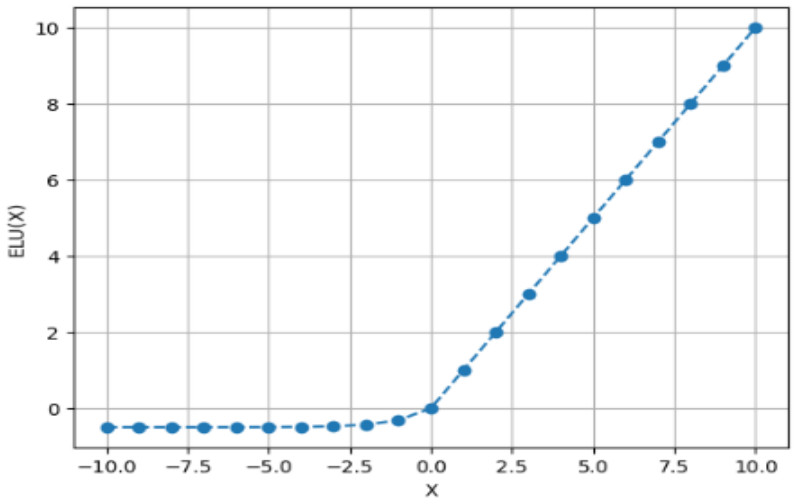
- Advantages: Simple, computationally efficient and has been shown to work well in practice. It avoids the vanishing gradient problem.
- Disadvantages: Not zero-centered and can suffer from the "dying ReLU" problem where the gradient becomes zero for negative inputs and stops training.

5. ELU Function:

[8]:

```
from math import exp
alpha = 0.5

elu = lambda x, alpha: x if x > 0 else alpha * (np.exp(x) - 1)
y = [elu(x_i, alpha) for x_i in x]
plot_graph(y,"ELU(X)")
```



- Advantages: Avoids the "dying ReLU" problem and can produce negative outputs. It is also differentiable for all inputs and has been shown to improve performance over ReLU.
- Disadvantages: Slightly more computationally expensive than ReLU.

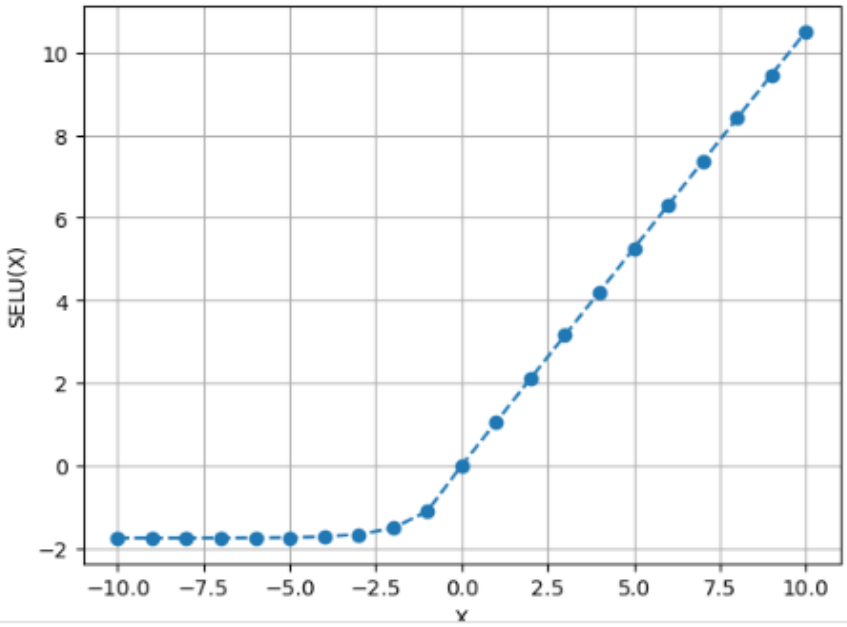
6. SELU Function:

[9]:

```
alpha = 1.67326
scale = 1.0507

selu = lambda x, alpha, scale: scale * (x if x > 0 else alpha * (np.exp(x) - 1))
y = [selu(x_i, alpha, scale) for x_i in x]

plot_graph(y,"SELU(X)")
```



- Advantages: Designed to achieve self-normalization in deep neural networks, leading to faster convergence and better performance. Zero-centered and has a smooth curve.
- Disadvantages: Requires proper initialization of weights and biases to achieve self-normalization.

Conclusion:

So, overall, there is no one-size-fits-all activation function that is better for all types of problems. It is best to try out different activation functions and select the one that works best for your specific problem and architecture. However, ReLU is a good default choice for many problems, and ELU or SELU can be good alternatives if negative inputs are present or deeper architectures are used.